

INTERNAL BUS SYSTEM FOR DFPs AND UNITS WITH
TWO- OR MULTI-DIMENSIONAL PROGRAMMABLE CELL
ARCHITECTURES, FOR MANAGING LARGE VOLUMES OF DATA
WITH A HIGH INTERCONNECTION COMPLEXITY

DI
BACKGROUND INFORMATION

FPGAs and DPGAs, and similar systems with two- or multi-dimensional programmable cell architectures, have internal bus systems which either have a global connection to all or most of the logic cells or have a local next-neighbor connection. Both types have in common the fact that they involve connections between two or more logic cells. In addition, always exactly one signal can use the bus, unless a multiplexer architecture is configured together with a control into a plurality of logic cells.

According to German Patent No. DE 44 16 881, the bus systems described above already exist in DFP-based systems. In addition, there is the possibility of separating a bus system, thus resulting in several sub-buses that can be used separately.

In German Application No. DE 196 54 595.1-53, an I/O bus system is described which combines a plurality of bus systems within the unit, where a unit may be an FPGA, a DPGA, a DFP, etc., and leads out of the unit. Memory modules or peripheral devices or other units of the generic type mentioned above can be triggered in this way. There is only one address register or address counter to generate the external addresses.

Conventional bus systems are not suitable for transmitting large volumes of data in the form of signals grouped by bytes or otherwise. Especially when the units are used for computation of algorithms, it is necessary

Express Mail: EM 171926708 ✓

09145139-002899

INS DI

for a plurality of data (packets) to be transmitted simultaneously between the individual configured function areas of a unit. In the usual technology, a direct point-to-point connection must be set up for each data path, which is the connection (the bus) between two (or more) function blocks that receive the same data, and which then regulates the data traffic between these function blocks exclusively. There can be only one data packet on the bus at a time. The interconnection complexity is very high. The transmission rate of today's internal buses is limited by the maximum bus size and the signal propagation time on the bus. With the I/O bus described in German Application No. DE 196 54 595.1-53, only one type of bus connection can be set up per I/O cell, namely exclusively that programmed in the address register. There is no possibility of responding to different types of data or data transmitters and connecting them to different external units.

SUMMARY OF THE INVENTION

An object of the present invention is to create a bus system that can transmit data between a plurality of function blocks, where multiple data packets can be on the bus at the same time. The bus system automatically recognizes the correct connection for various types of data or data transmitters and sets it up.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a two-dimensional unit with a cell array and nodes.

Figure 2 shows further detail of Figure 1.

Figure 3 shows several existing bus connections and the setup of new connections.

Figure 4 shows the setup of new bus connections.

Figure 5 shows the next step in setting up the connection.

Figure 6 shows the collision of two buses.

5

Figure 7 shows the stepwise clearing of a connection after a collision.

Figure 8 shows the stepwise clearing of a connection after a collision.

10

Figure 9 shows the stepwise clearing of a connection after a collision.

Figure 10 shows the stepwise clearing of a connection after a collision.

15

Figure 11 shows the resetting of the cleared connection after a certain period of time.

20

Figure 12 shows the continuation of Figure 6 when the node has more than one busbar and thus there is no collision.

Figure 13 shows the connection of the bus segments to the busbars of a node.

25

Figures 14a-g show the status of a data transfer of a connection being set up at intervals of one bus cycle.

30

~~Figures 15a-h show the status of a data transfer of a connection release at intervals of one bus cycle.~~

Figure 16 shows the control unit of a node.

35

Figure 17 shows an exemplary broadcasting to several data receivers.

09145130-082800

Ins
EI

Figure 18 shows another embodiment of the control unit shown in Figure 16, for a node with collision detectors.

Figure 19 shows another embodiment of the routing table shown in Figure 16, with the control of Figure 18.

Figure 20 shows a busbar that can be used in Figures. 18 and 19.

Figure 21 shows a further exemplary broadcasting to several nodes.

Figure 22 shows the sequence control for the devices shown in Figures 18-21.

Figure 23 shows a bus node according to the extended routing method.

Figure 24 shows an optimized variant the bus node of Fig. 23.

Figure 25 shows a data register with a synchronization unit.

Figure 26 shows a collision detector according to the extended routing method.

Figure 27 shows a control unit for Figures 23-26.

Figure 28 shows a flow-rate-enhancing, cascadable addition to Fig. 27.

Figure 29 shows the control register for the device of Figure 27.

Figure 30 shows a bus connection setup via relative spatial coordinates according to the extended routing

09145139 082898
358280" 5ET460

method.

Figure 31 shows a bus connection via absolute spatial coordinates according to the extended routing method.

Figure 32 shows the management of spatial coordinates.

Figure 33 shows a segmented bus setup.

DETAILED DESCRIPTION

A bus system according to the present invention can be integrated on a unit horizontally, vertically, diagonally or in any other position. The bus system is subdivided into a plurality of segments, with the segments being separated by a bus control circuit. This bus control circuit is called a node. At the same time, the node is responsible for routing, i.e., controlling the direction of flow of the data. Logic cells or PAE cells according to German Application No. DE 196 51 075.9-53 are connected to the nodes; they send and receive their data over the nodes, and a target address can be delivered along with each data packet. Likewise the bus system is capable of generating target addresses via look-up tables, referred to as routing tables below; the method described in German Application No. DE 196 54 846.2-53 serves this purpose in particular. The bus systems are especially suitable for direct connection to the INPUT/OUTPUT CELLS described in German Application No. DE 196 54 595.1-53. Connections are set up and data is synchronized automatically. If a connection fails because the required bus segment is busy at the moment, the connection is attempted again at a later time. Different connections can be set up automatically on the basis of different types of data or data transmitters.

INS
D1

09145139 082299

There are a plurality of buses arranged horizontally, vertically, diagonally or in any position on the unit. The individual buses do not lead continuously from one edge of the unit to the other, but instead they are subdivided into a plurality of bus segments. The bus segments are separated by the nodes described below. Each bus segment can be used and connected independently of the others, with the interconnection being the responsibility of the nodes. A separate protocol that is administered by the nodes can be implemented on the bus system. It is also conceivable for the protocol to be administered by the cells using the bus and for the nodes to be only passive switches.

The node

A node provides the interconnection among individual bus segments. Connection of the cells to the bus segments is also handled by the nodes.

The bus segments of all directions are combined in a node; this means that:

- in a two-dimensional system, the buses are run to the node from four directions: north (N), south (S), east (E) and west (W);
- in a three-dimensional system, the buses are run to the node from six directions: north (N), south (S), east (E), west (W), top (T) and bottom (B);
- in an n -dimensional system, the buses are run to the node from n directions (for each dimension there is a directional vector whose direction is determined by its plus or minus sign \Rightarrow for each dimension there are two directions which are differentiated by the sign in front of the directional vector).

A-type node

09145139 082838

Within the node there is a bus system to which the external buses are connected and which thus consists of several busbars. A first external bus is connected to a busbar across a gate. The busbar is connected to the second external bus via another gate. To permit broadcasting, i.e., sending data to multiple receivers, several "second" buses can also be connected to the internal bus system. The gates may be embodied as purely passive switches, bus drivers, or registers/latches.

In addition, the node has access to one (or more) configurable elements (cells) connected to it. It optionally connects them to one or more adjacent bus segments.

B-type node

In contrast with the A-type node, there is no internal bus system. The node has n configurable switches, which can connect each neighboring segment to each of the other neighboring segments. For example, for a 2-dimensional structure, n=6.

N	Connection
1	E-W/W-E
2	N-S/S-N
3	N-E/E-N
4	E-S/S-E
5	S-W/W-S
6	W-N/N-W

(N=north; E=east; S=south; W=west)

A switch may be unidirectional or bidirectional; it may contain a register or latch for storing data.

The "regular" routing method

5 A connection is initiated by a data transmitter (DS) -
this is a configurable element (logic cell, bus cell
(including external cells according to German Application
No. DE 196 54 595.1-53)) - which needs a connection to a
data receiver (DE) - which also includes a configurable
10 element. The DS informs the node of its bus request. The
first node directly downstream from the data transmitter
is called the initiator node. It takes the address of the
next node needed for the data transfer from an internal
table, as described below. If the node is capable of
addressing the first bus segment needed - this is always
possible when the bus segment **AND** an internal busbar of
15 the node are free - it sets up the connection; otherwise
it rejects the request by the DS, and the DS attempts to
gain access again at a later time, or it waits and
maintains the access request until the node accepts it.

20 Each successive node in turn takes the address of the
next node from its internal table and continues to set up
the connection. If the node is not capable of setting up
the connection (either the required bus segment is busy
or the busbar of the node is not free), it can either
jump to a waiting state or interrupt the connection and
25 report back an error to the initiator node.

30 Only when a connection has been completely set up are the
data packets transmitted and the transmitter receives the
acknowledgment signals for the data exchange (see rACK in
German Application No. DE 196 51 075.9-53). The data
transfer is thus automatically synchronized with the data
transmitters. If the connection is not set up and must be
attempted again at a later time, the data transmitter
35 does not receive an acknowledgment signal, so no data is
lost.

Once the connection has been completely set up, it

remains quasi-permanently (i.e., it appears to the DS and DE like a direct connection) until the DS breaks the connection again by sending a message to the initiator node. It is conceivable to implement a timeout process which interrupts a standing connection after a certain period of time, in particular when no data transfer has taken place for a long period of time, to clear the bus segments for other connections.

The extended routing method

In the previously described routing method, the addresses are on the bus only during the routing phase. During data transmission, the addresses are no longer transmitted. In the extended routing method, the addresses always run on separate lines together with the data. Two addressing schemes exist:

1. Spatial coordinates

The spatial coordinates of the target are given as addresses. The spatial coordinates depend on the selected system dimension. For example, a 3-dimensional system uses the X/Y/Z coordinates, while a 2-dimensional system uses the X/Y coordinates. In principle, the method can be used in a system of any number of dimensions. A directional vector containing the information of whether the data/routing moves in the positive or negative direction is assigned to each coordinate.

The data moves from the source node in one of the directions set. The directional coordinate is modified when passing through a node so that

- when moving in the positive direction, 1 is subtracted;
- when moving in the negative direction, 1 is added.

The target axis is reached when the coordinate is 0; the target node is reached when all coordinates are 0.

It depends on the implementation whether a complement of

two is generated and a coordinate is represented as a negative number (on which addition is performed) or whether subtraction is performed on a positive number. Addition can be performed on a positive number until an overflow is generated, which indicates the target position.

There are two rational strategies to determine the direction of data transmission:

a. Static method: The data always runs in the same direction; i.e., an attempt is made always to preserve the direction until a change in direction becomes unavoidable. A change in direction is unavoidable when the current directional coordinate is zero, i.e., its target position has been reached. If the target position of a coordinate has been reached, the data moves in the direction of the non-zero coordinates. If all the coordinates are zero, the target node of the data has been reached.

b. Dynamic method: The data moves in any of the possible directions, always in the direction given by the directional vector. "Moving in any direction" means that the data is always forwarded to the node receiving the least traffic. Thus, an attempt is made to select the path of fewest collisions and blockages. This may make the process faster and more appropriate in the case of large systems.

2. Address lookup

If lookup addresses are transmitted, the next node is determined in each node. This takes place by reading the lookup address in the routing tables described below, and thus determining the data for the next target. This process is the same as in the "regular" routing method.

The advantage of the spatial coordinates is that no table

09145139 "082090

lookup is needed, which reduces the overhead.
The advantage of the address lookup is its flexibility
and the exact predictability of the connection.

5 Sometimes it can be advantageous to mix the two methods.
To do so, both addresses (spatial coordinates and lookup
addresses) must be transmitted simultaneously. If the
spatial coordinates are not zero, the data is transmitted
using the spatial coordinates. As soon as the spatial
10 coordinates reach zero, a lookup is performed in the
current routing table at the site of the lookup address.
This makes it possible to set up, in a flexible manner,
the segments over which the data is transmitted by the
lookup method or the spatial coordinate method.

15 **Bus states of the extended routing method**

1. Quasi-permanent

A connection can be set up quasi-permanently in a manner
20 similar to the "regular" routing method. Each node
through which a first data passes is permanently enabled
according to the addresses associated with this data.
This enable is maintained for all the subsequent data
until the connection is broken. Although such a setup is
25 not essential in the extended routing method, it offers
two advantages:

- i. the transit time of the data is considerably reduced;
- ii. time is not wasted in arbitration.

30 **2. Registered**

This is the normal bus status. Incoming data is buffered
in a register. When writing into the register, an
acknowledge signal is sent to the transmitting node. Data
is only written into the register when the latter is
35 empty, i.e., either no data has been written into the
register or the previously written data has already been
retransmitted. If the register is not empty, a wait

Each line represents a valid connection. Several connections may be active at the same time, the maximum being as many connections as there are free internal busbars and free external bus segments. A monitoring logic unit which is described below is responsible for testing whether a newly selected connection can be set up.

Each line is addressed and selected by its binary address. In addition, there are special lines that do not have a binary address, but instead are selected by trigger signals or status signals. These signals include (see German Application No. DE 196 51 075.9-53):

- rRDYl, rRDYh
- oACK1, oACK2

Bus connections to the data receivers (rRDY) are automatically set up here whenever the data transmitter cell has valid results. With the oACK signals, the receiver/transmitter sequence is reversed. The data receiver sets up the connection with its data transmitter as soon as the data receiver has processed its operands and is capable of processing new operands.

The active gates for the respective connection are entered in the Gate columns; it is possible for only one gate to be marked and one connection to a configurable element (one or more cells), e.g., to a R-REG or O-REG to be selected (see German Application No. DE 196 51 075.9-53) in the *EALU* columns.

In the *Bus* column, the internal busbar used for this connection is selected; this value is binary so that a total of four internal busbars can be used in the table described here. Selection of the internal busbars can also be omitted if a priority decoder recognizes the first free busbar and automatically allocates it.

The column *Entry position in the target table* shows the address of the line of the table that controls the node to which the connection is to be set up. The routing information in the next node needed for the current connection is at this address.

The *Run time* column is optional. The expected signal propagation time from DS to DE can be given there. This information can be used for computation of the data throughput or for generating a timeout.

The *Setup time* column is optional. The maximum time for setting up the connection to the next node (or the entire connection from DS to DE) can be entered there. If this time is exceeded, the connection can be terminated by a timeout and the bus segments and nodes become free for other connections. At a later time, DS will again attempt to set up the connection.

Entries in the routing table can be configured and reconfigured by a program loading unit according to conventional methods (See German Application No. DE 196 51 075.9-53 and German Application No. DE 196 54 846.2-53).

If the "extended" routing method is used, the spatial coordinates must be entered in the routing tables. A priority identifier is also expected.

The priority identifier provides the information on the relevance of a channel for the performance of a module. The higher the priority identifier, the higher the relevance of the channel. The identifier can be configured in three different manners:

1. Timeout

The identifier shows after how many unutilized clock

cycles should the channel be cut off, i.e., after how many cycles is a DISCONNECT generated.

2. Packet size

5 The identifier shows after how many data packets is a DISCONNECT generated.

3. Clock cycles

10 The identifier shows after how many clock cycles is a DISCONNECT generated.

One of the three types can be permanently implemented or a type can be selected via additional information.

15 The priority type (*priotype* in the table below) is evaluated in the routing table given below as an example:

09145139 082200
TO 160

Bit Combination	Priority Type
00	Timeout
01	Packet size
10	Clock cycles
11	Permanent

20
25 Permanent can also be represented by permanently providing one value of the priority identifier as "permanent." The maximum value of the priority identifier or zero is particularly suitable for this.

T0170

5

Gate				EALU				Bus		Entry Position in Target Table	Y/X Space Coordinate & Directional Vector	Priority Identi- fier	Prio- rity type	
1	2	3	4	O	O	R	R	1	0	k - 0	m - 0	n - 0	1	0
				1	2	1	2							

The arbiter

10 Upstream from the routing table is an arbiter, which
selects a few signals via the routing table from the set
of requests for setting up a connection. It is
appropriate to set up the arbiter from a conventional
priority logic unit and a conventional round-robin
15 arbiter (it always switches the highest priority on a
time segment to the next signal, i.e., the signal having
the highest priority at the moment will next have the
lowest priority and then will have a higher priority with
each access). The priority logic unit can serve to
20 allocate an especially high (or especially low) priority
to some signals, e.g., rACK, oRDY. The round-robin
arbiter ensures that a requested connection which cannot
be set up at the moment will have the lowest priority and
will thus wait until all other connection requests have
25 been either set up or reviewed.

The state machine (control)

A state machine controls the internal sequences in the
node. The state machine is divided into two parts:
30 1. Control of the node,
2. Control of bus transfer and synchronization. A state

09145139-082898

machine like that described in DE 196 51 075.9-53 may be used.

The implementation of state machine(s) are known to those skilled in the related art and thus, it will not be described in greater detail here.

The monitoring unit

A monitoring unit is connected downstream from the routing table. It takes the data entered in an addressed line and checks on whether it is possible to set up a connection. The following points in particular are checked:

- Is a free internal busbar available?
- Is the requested gate free?
- Is the requested external bus segment free?
- a) If the check is positive, the ACCEPT signal is generated and sent to the state machines as well as the unit requesting that the connection be set up to signal successful connection setup.
- b) If the check is negative, the REJECT signal is generated and sent to the state machines as well as the unit requesting that the connection be set up to signal a failed connection. The arbiter described above can respond to this signal and set the priority of this request at the lowest level.

Breaking a connection

An existing connection can be broken by various criteria. The most important criteria include:

- **Timeout:** A connection is broken because no data transfer has taken place for a long period of time. A timeout can be implemented easily by a loadable decrementer. With each data bit transmitted, the counter is loaded again with a fixed starting value representing the period of time until timeout. If

there is no data transfer, it is decremented by one with each bus clock pulse. When it arrives at zero, the maximum period of time has elapsed and the bus is disconnected.

5

- **Data counter:** A loadable counter is loaded with the number of data bits to be transmitted. With each data bit transferred, the counter is decremented by one. If the counter arrives at zero, all the data has been transmitted and the bus is disconnected.

10

- **Synchronization signals** (see German Application No. DE 197 04 728.9, RECONFIG): The bus disconnect is controlled by the status and/or synchronization signals of the cell(s) functioning as data transmitters. For example, if the cells indicate that their data processing is concluded or that they can be reconfigured, the bus is disconnected because it is no longer needed. These signals also include rRDY and oACK.

15

20

The clearing of the connection proceeds by a signal being sent from the initiator node to break the connection. Each additional node sends the received signal on to its partner node(s) and breaks the connection immediately.

25

In the extended routing method, quasi-permanent buses are disconnected either by the RECONFIG sync signal (see German Application No. DE 197 04 728.9) or according to the priority identifier. If the end of a connection is determined by a node on the basis of the priority identifier, the node generates the DISCONNECT signal to disconnect the bus and forwards it to all the other nodes. The nodes respond to a DISCONNECT as they do to a RECONFIG.

30

35

In register-oriented nodes, disconnecting is not

necessary, since a connection is set up dynamically with the incoming data according to the assigned addresses. After the data is forwarded, the connection is automatically broken and becomes free for other transmissions.

Broadcasting

German Application No. DE 196 51 075.9-53 describes a bus system that permits transmission of a data packet to multiple receivers and also allows an acknowledgment of the data transfer. The same system can also be used in accordance with the present invention. It is possible to readily switch several gates to one busbar. Only one line in the routing table is used. The address within the target routing table *Entry position in the target table* must necessarily be the same in each of the selected routing tables.

To circumvent this problem, it is possible to make available multiple entries as *Entry positions in the target table*. For example, a separate *Entry position in the target table* could be available for each gate. Thus, an address within the target routing table is assigned to each gate.

The acknowledgment takes place exactly as described in German Application No. DE 196 51 075.9-53 over signal lines driven by open-collector drivers and terminated with a transistor.

In order to provide better implementability in existing chip technologies, acknowledgment signals in a node can be first masked and then logically gated. The logically gated signal is then forwarded to the next node. If, for example, a logical AND-gating of the acknowledge signals takes place in all the nodes on the path, the result is

the same as in the case of an open-collector circuit.

Program loading unit (PLU)

A program loading unit as described in German Patent No. DE 44 16 881 A1 is connected to the routing table. By means of this PLU, the routing table - equivalent to the switching tables in German Application No. DE 196 54 846.2-53 - can be configured and reconfigured. The routing tables can be expanded by one column which is used to send feedback to the PLU when the connection indicated in the respective line is set up or cleared. Thus, it can be indicated in this column whether feedback is to go to the PLU when the connection is set up or cleared and if so, what type of feedback. Feedback is sent according to DE 196 51 075.9-53, as shown in Figure 15, where instead of latch 1504, a gate is used which is switched through when the connection is set up or cleared, depending on the setting, and triggers the feedback to the transistors (1502) which are wired as open-collector drivers.

Remaining table	PLU		
	Message when setting up connection	Message when clearing connection	Binary value of the feedback
...	b	b	2 1 0

Figure 1 shows a unit which may be an FPGA, DPGA, or a DFP. (See German Patent No. DE 44 16 881 A1). The unit is composed of configurable cells (0101) and is symmetrical in two dimensions; 0101 may also represent a plurality of

configurable cells (including different cells) that are combined in a group and interconnected. The nodes of the bus system (0102) are between the cells. Several nodes are shown, which will set up several connections in the following example. Data transmitter A (0103) will set up a connection to data receiver A (0104), and data transmitter B (0106) will set up a connection to data receiver B (0105). Figure 2 shows an enlarged detail (0107).

Figure 2 shows a detail of the generic unit described here. The configurable cell(s) from Figure 1 (0101) are shown as 0201. A bundle (0203) of several lines (any number is possible and is not shown exactly) connects 0201 to a node 0202. The nodes (0202) are interconnected over the bus segments (0205). The nodes are also connected over bus segments 0206, configured as bus segments 0205, to the nodes outside the detail. The optional line bundle (0204) illustrates that the configurable cell(s) may also be connected to multiple nodes (0202) over several different line bundles.

Figure 3 shows the unit at run time. There are several connections:

- between nodes Z (0301)
- between nodes Y (0306)

Data transmitter A (DSa) (0302) attempts to set up a connection (0303) to data receiver A (DEa) (0307).

However, the connection is rejected (REJECT) because it is blocked at node Z (0308). At the same time, data transmitter B (DSb) (0304) sets up a bus segment (0305) to its receiver. This attempt succeeds because the node addressed and the required bus segment are not blocked.

Figure 4 shows the next bus cycle. Connections Y and Z have been cleared in the meantime. Data transmitter A

(0401) can now set up the bus segment (0403) because the node (0402) is no longer blocked. At the same time, data transmitter B (0404) expands the existing bus segment (0405) beyond the node (0406) \Rightarrow the new bus segment (0407) is set up.

Figure 5: Setting up the bus which was begun in Figure 3 and continued in Figure 4 is continued in a manner equivalent to that in Figure 4.

Figure 6: The attempt to set up a connection of bus segment 0602 for data transmitter B fails. Node 0601 is busy and delivers to node 0603 a REJECT signal that signals the failure of the attempted connection and causes the connection to be terminated.

Figure 7 shows the complete connection set up between data transmitter A (0701) and data receiver A (0702). The connection of data transmitter B is further cleared. Node 0703 delivers the REJECT signal to node 0705 over the bus segment (0704). Then segment 0704 is cleared.

Figure 8: The data transfer between data transmitter A and data receiver A begins. The connection of data transmitter B is further cleared. Node 0801 delivers the REJECT signal to node 0803 over the bus segment (0802). Then segment 0802 is cleared.

Figure 9: The data transfer between data transmitter A and data receiver A continues. The last segment of data transmitter B is cleared. Node 0901 delivers the REJECT signal to node 0903 over the bus segment (0902). Then segment 0902 is cleared.

Figure 10: The data transfer between data transmitter A and data receiver A continues. Data transmitter B (1001) waits a period of time until it again attempts to set up

a connection to its data receiver.

Figure 11 shows the status several bus cycles later. The data transfer between data transmitter A and data receiver A is still taking place. Data transmitter B (1101) starts a new attempt to set up the connection to its data receiver. It sets up a bus segment (1102) to node (1103). If the connection of data transmitter A to its data receiver A is cleared in the next bus cycles, the connection of data transmitter B (1101) will be set up successfully; otherwise it will fail again as described in Figure 6.

Figure 12 shows the continuation of Figure 6 when node 1202 is capable of setting up more than one connection, i.e., the node has several internal busbars. Thus the connection of data transmitter A can be handled over the first busbar, and the connection of data transmitter B can be handled over the second busbar. Node 1202 sets up bus segment 1203 to data receiver B (1201).

Figure 13 shows an interconnection structure within the node. There are four busbars 1301, 1302, 1303, 1304 within the node. The busbars are switched to the bus segments west (1318), north (1316), east (1319), south (1317) over one group of gates each (1308, 1309, 1310, 1311). The busbars are connected to O-REG1/2 (see German Application No. DE 196 51 075.9-53) over gates 1307 (1314, 1315). The R-REG (see German Application No. DE 196 51 075.9-53) is connected to the busbars over gates 1306. The lower-order and higher-order R-REG (1312, 1313) can be wired separately. The gates are driven via bus 1320. The required gate system (west, north, east, south) and the internal busbar are indicated. The required gate (1321, 1322, 1323, 1324) is selected by AND-gating (1325, 1326, 1327, 1328) the directional information with the information on the busbar selected.

09145139-082898

Figure 14a shows the setting up of a connection, with the data transmitter (1401) transmitting the first data packet to the node of the data transmitter (1402).

5 Figure 14b: The node selects from the routing table the entry belonging to rRDY - rRDY is a status signal which indicates that data is ready at the data transmitter (see German Application No. DE 196 51 075.9-53). On the basis of this entry, the next bus segment is set up and the
10 address of the routing table of the next node is transmitted.

Figure 14c: The last node (1403), the data receiver node, receives the address for the entry within its routing
15 table. The entry points to a cell, not to another node. Therefore, the node immediately activates its gates to the selected cell.

Figure 14d: The data goes through the activated gate of
20 1403 directly to receiver cell 1404.

Figure 14e: The cell sends back the oACK signal (see German Application No. DE 196 51 075.9-53) as an acknowledgment for the data received. In the next bus
25 cycle (see Figure 14f) the data transmitter will then send the next data packet.

Figure 14f-g: Normal data exchange is taking place
30 between the cells.

Figure 15a: There is a data connection from the data transmitter (1501) to the data receiver (1503) over several nodes.

35 Figure 15b: The data transmitter (1501) has terminated its data transmission and is sending a DISCONNECT signal to the first node.

Figure 15c: The first bus segment is cleared and the node relays the DISCONNECT signal further.

Figure 15d: The clearing of the connection proceeds.

5

Figure 15e: The last node receives the DISCONNECT. Then the last node clears the connection to the preceding node and to the data receiver at the same time.

10

Figure 15f: The last bus segment and the connection to the data receiver are cleared.

15

Figure 15g shows a disconnect process where the DISCONNECT line is looped through all nodes. The DISCONNECT is propagated in one clock cycle; all segments are disconnected at the same time.

Figure 15h corresponds to Figure 15e.

20

Figure 16 shows the control of a node. Requests (1601) for setting up a connection are transmitted over the priority decoder (1602) to the routing table (1603). The priority decoder selects the request with the highest priority, with the request that has just failed being assigned the lowest priority. The priority logic unit receives its requests over status signals (e.g., of the configurable cells (see German Application No. DE

25

196 51 075.9-53) rRDY, oACK) or over the bus segments 1316, 1317, 1318, 1319. If data is applied to the bus

30

segments without the gate of the bus segment being activated, the data is interpreted by the priority logic unit as the address of the routing table and is regarded as a request. If status signals are applied to the

35

priority logic unit (rRDY, oACK), they are transmitted in addresses for the routing table. The addresses of the routing table select an entry. The data of the entry (1604) is relayed to an AND-gate unit (1605). The binary

number of the bus selection (BUS 1-0) is transmitted to select signals via a 2:4 decoder (1606). The AND-gate unit AND-links each signal with the same signal stored in a latch (1607). This means that the GATE1 signal of the routing table is AND-gated to the GATE1 signal of the latch, the GATE2 signal of the routing table is AND-gated to the GATE2 signal of the latch, etc. The signals of the latch represent the status of the instantaneous interconnection structure, i.e., the gates used and the busbars being used are entered into the latch. If an AND-gating of the connection request to the current status yields a true level, this means that the new connection request requires resources that are in use at the moment. All the AND-gates are OR-gated with one another (1608). If a true level is obtained at the output of the OR-gate, the connection request is rejected (REJECT) (1609), because the required resources are busy. The ACCEPT signal (1611) is generated from the REJECT signal over an inverter (1610). The signals are relayed to a state machine (1612) which can be implemented by a known technology. This then controls the acceptance or rejection of the connection. If the connection request is rejected, the state machine sends the REJECT to the priority decoder via 1613 and the request is assigned the lowest priority. If the request is accepted, the new status signals are OR-gated with the current status signals after the latch (1614) (the OR-unit has the same design as the AND-unit (1605) described above) and written back into the latch (1607). The state machine controls via 1623 whether the OR-unit (1614) or the mask (1616) is active. The latch is triggered by the state machine via 1622. The new setting goes to the gates via bus 1615.

A bus connection is cleared by a similar procedure. However, the REJECT signal must occur in a check of resources, because the bus connection that is to be

cleared must exist. On the basis of the REJECT, state machine 1612 activates mask 1616 instead of OR-unit 1614. The connection data on the connection to be cleared is masked out of the current interconnect status and written
5 back into latch 1607. Before rewriting the new connection data, the state machine sends the DISCONNECT signal to clear the connection to the next node.

The control can directly access busbars 1301, 1302, 1303,
10 1304 over gates 1617, 1618, 1619, 1620. Thus, the state machine can transmit control signals (DISCONNECT) to the busbar and can also receive control signals (REJECT, ACCEPT) from the busbar and can respond to them. Likewise, these gates serve to transmit the *Entry*
15 *position in the target table* (via 1621) to the busbar.

The program loading unit (PLU) can access the routing table via 1624.

Figure 17 shows broadcasting of a data transmitter (1701) to multiple data receivers (1702) over a plurality of nodes (1707), which are not discussed further here. The bus is shown as divided for the sake of illustration, namely, divided into the acknowledge line (ACK) (1703)
20 and the remaining bus (1704). ACK is negated and supplied to the inverting open-collector bus driver. ACK is pulled to H level over a pull-up resistor 1705. The circuit is designed so that the following cases occur:

30 -If the corresponding bus is not activated, L is applied to the base of the transistor (1706). Thus it does not use the bus.

35 -If the corresponding bus is activated and the signal is not acknowledged, H is applied to the base of the transistor (1706). This means that the bus is pulled to L. If a result of broadcasting is distributed to multiple

data receivers, all nodes that have not yet acknowledged the result data and need waiting cycles pull the bus to L.

-If the corresponding bus is activated and the signal is acknowledged, L is applied to the base of the transistor (1706). This means that the bus is not used. If a result is distributed by broadcasting to multiple data receivers, then none of the nodes that have acknowledged the result data and do not need waiting cycles use the bus.

Since the bus in its initial state assumes the H level, i.e., the acknowledgment, non-acknowledgment according to case 2 overrides acknowledgment by pulling the bus to L. The bus does not go to the H level, i.e., the acknowledgment status, until all nodes involved in the connection have sent an acknowledgment. Thus, a wired-AND circuit is implemented.

An example of implementation is illustrated for a node having the following bus architecture:

Data	Data Signals
ACK	Data and connection handshake
RDY	Data handshake (data present)
SETUP	Connection handshake (the segment is set up)
DISCONNECT	Disconnect request (through timeout)
RECONFIG	Disconnect request (through reconfiguration of the configurable cells involved)

Figure 18 shows the control of the busbars of Figure 13. Contrary to the embodiment described previously, where

the busbar is predefined in the routing tables, the logic is independently capable of determining and assigning a free busbar.

5 Each trigger unit 1801 is assigned to a busbar. A trigger unit made up of a gate (1801a), for sending the address information of the routing table to the node connected to the busbar, and a register (1801b) that is responsible for driving the busbar. 1801 receives, via bus 1813, the
10 data, via a circuit described in Figure 19, from the routing table. The gates with access to the respective busbars are enabled via 1801b. Each gate has an enable signal assigned to the entry in 1801b. When no entry is set, the respective busbar is free and can be assigned to
15 an access request. The check is performed by an OR function (1802) via all the enable signals sent to the gates. The results of 1802 from all busbars are sent to an arbiter (1803), which selects one of the free busbars and addresses its 1801 via an address bus (1804). If no
20 busbar is free, 1803 signals this via 1805 of the controlling state machine. An entry in 1801b identifies a gate assigned to the busbar. The position in each 1801b is the same. This means that the enable signal for gate p starts at position p of 1801b, the enable signal for gate
25 p+1 starts at position p+1, and the enable signal for gate q starts at position q. If an OR function (1806) is executed via the enable signal of a gate p, the result provides the information of whether or not gate p is free. There is such a checking function for each gate
30 (1807 = gate p+1, 1808, 1809 = gate q). All gates for which the current connection setup is irrelevant are masked out via a mask 1810, i.e., irrelevant gates are forwarded by the mask as disabled. An OR function determines whether one of the gates is enabled. Since all
35 the irrelevant gates are marked as "disabled," only the status of the gate that is necessary for the current setup is forwarded to the state machine (1812). If the

desired gate is enabled, it cannot be used in the current setup, since otherwise a collision would occur. Connection setup is aborted and is either rejected or attempted later.

5

Figure 19 shows the routing table and the arbiter for selecting the active bus. Each bus (2004, 2005, 2006) connected to a node sends an access request via a signal (1901) to an arbiter (1902), which selects one of the access requests. The multiplexer (1904) is activated by a decoder (1903) so that either the number of the selected access (through a configurable cell for direct access) (1905) or the lookup address of the selected access is forwarded to routing table 1906. 1906 outputs the data assigned to the value of 1905. The lookup address of the next node is transmitted via 1813 to 1801a. The address of the next node, decoded in the decimal form (1907), to 1801b via a mixer (1908), usually made up of OR-gates, to 1801b over bus 1813. If the bus to the next node is set up, the gate to the previous node can be enabled by switching the addresses of the previous node, decoded by decimal decoder 1909, in mixer 1908 to bus 1813 and transmitted to 1801b.

20

25

Figure 20 shows the busbars (2001, 2002, 2003) for connecting the adjacent buses (2004, 2005, 2006). The buses are connected to the busbars over the multiplexers/demultiplexers (2007), triggered by 1801b; a multiplexer p receives the output signals p of all 1801b; a multiplexer p+1 receives the output signals p+1 of all 1801b, etc. The individual signals represent adjacent busbars, since each busbar control controls exactly one (p) of a plurality of signals (p_x, p_{x+1}, \dots, q_x). If a signal assigned to a busbar control is set, the corresponding busbar is connected through a 2007.

30

35

The timeout generators (2008) control the disconnection

of the respective segment and the connected bus. The timeout generators are configured directly by the routing table (1906). The connection is not shown in the figures for the sake of clarity.

5

The 1801a's assigned to a busbar are connected to this busbar.

10 In Figure 21, a bus segment (2105) is broadcast to several nodes (2102, 2103, 2104) via one node (2101). The RDY handshake is transmitted, like the data, directly to each receiver. The response ACK handshakes are sent to the OR-gate (2107) and the AND-gate (2108) via masks (2105, 2106). The relevant ACK and whether an ACK is
15 forwarded through a Boolean AND-function or an OR-function is selected via the masks. Both functions are combined over an OR-gate (2109). If an ACK is irrelevant, mask 2105 forwards a logical 0 (L level), while mask 2106 forwards a logical 1 (H level). Masks 2105 and 2106 are
20 set separately by the routing table. The connection is not illustrated in the figures for the sake of clarity.

Figure 22 shows the state machine of the circuit described.

25

The base state is "IDLE," which is not exited by the state machine until a "request" (access) has appeared AND a busbar AND the selected gate are free. Then the state machine acknowledges the bus setup to the previous state
30 machine by sending an ACK handshake. The state machine goes into the SEND mode, during which the data of the routing table is sent (via 1801a) to the next routing table. The state machine exits this mode only when it receives an ACK handshake from the next routing table OR
35 when a "disconnect" signal is received by the bus, e.g., due to a timeout. In the event of a "disconnect," the state machine goes into the DISCONNECT mode to disconnect

09145139 032898

the bus (this mode is not essential; in another exemplary embodiment, the state machine may jump back to IDLE, but this example illustrates the operation better). When an ACK handshake is received, the state machine jumps back to IDLE; the gate of the previous routing table is enabled in 1801b via 1909/1908. In order to improve the performance of the routing table, waiting for an ACK handshake during the SEND mode may be omitted. For this purpose, the access data to be sent to the next routing table should be buffered in 1801a during the SEND mode, i.e., 1801a is a register; at the same time, the information on the previous routing table should be written into an additional 1801b during SEND. Receipt of the ACK handshake of the following routing table causes 1801a to be switched off in the logic set up independently and switches from the first to the second 1801b, by storing the wiring of the gate of the previous routing table.

The following figures describe an example of implementation according to the "extended" routing method; the bus has the following configuration:

Data	Data signals
ACK	Data and setup handshake
RDY	Data handshake (data is present)
DISCONNECT	Bus disconnect request (via timeout)
RECONFIG	Bus disconnect request (via reconfiguration of the configurable cells involved)
X/Y	X/Y Spatial coordinates
LUT	Lookup address for the routing table
ADR-MODE	Shows whether the X/Y spatial coordinates or the lookup address should be used. If X=0 AND Y=0, the lookup address is automatically used.

Figure 23 shows a node with switching elements 1, 2, 3,

4, 5, 6, as well as buses B1, B2, B3, B4. The figure shows a single-line system for greater clarity. In fact, it is a bus system if only all the figures are taken as many times as there are buses. The switching element is, in the simplest case, a transistor (2301) (Figure 23b). In order to buffer data, the switching element is expanded with one register (2302) as shown in Figure 23b. The bidirectional buses Bn and Bm are connected to the register via transistors T1, T2, T3, T4 so that either the Bm->Bn transition or the Bn->Bm transition is stored. As an alternative, data can be transferred bidirectionally without buffering via T5. The mode of the switching element is set via control signals S1, S2, S3, S4 as follows:

S1	Bm->Bn (unidirectional, stored)
S2	Bm<->Bn (bidirectional, not stored)
S3	Bn<->Bm (bidirectional, not stored)
S4	Bn->Bm (unidirectional, stored)

Figure 23c: In order to obtain better signal quality, Schmitt input triggers and output drivers (2303) are used. The drivers (2303) are designed so that either the output driver or the input driver is enabled via a control signal depending on the level. Bidirectionality is lost; only unidirectional procedures can be implemented.

The mode of the switching element is set as follows:

S1	Bm->Bn (unidirectional, stored)
S2	Bm->Bn (unidirectional, not stored)
S3	Bn->Bm (unidirectional, not stored)
S4	Bn->Bm (unidirectional, stored)

Figure 23d: The inputs and outputs are sent via different

T0340

T0341

conductors (Bmi, Bmo, Bni, Bno) for better implementation in chips. The drivers (2304) are unidirectional. Driver control may be omitted.

5
T0350

S1	Bmi->Bno (unidirectional, stored)
S2	Bmi->Bno (unidirectional, not stored)
S3	Bni->Bmo (unidirectional, not stored)
S4	Bni->Bmo (unidirectional, stored)

10 Figure 24 shows a node similar to the node of Figure 23. The advantage of the node consists of its better implementability and register management. The drawing shows a single-line system for greater clarity. In fact, it is a bus system if only all the figures are taken as
15 many times as there are buses. The registers and drivers (A, B, C, D) are upstream from switching elements (1, 2, 3, 4, 5, 6). The switching elements are reduced to the cross-connection shown in Figure 24a. The inputs (Imi, Ini) are selectively connected to outputs (Imo, Ino) with
20 the help of control lines S5, S6 by transistors T6, T7.

Figures 24b-d show different configurations of the registers and drivers (A, B, C, D).

25 Figure 24b shows a bidirectional bus connected either as the input to register 2401 via T8 or to the output via T9; the bus within the node, Imo, represents the signal source. T8 and T9 are controlled via control line S7. A register bypass can be connected using the transistor
30 pair T10/T11, controlled via S8, to enable the quasi-permanent mode. The output of the register goes to bus Imi within the node. Imi and Imo are interconnected via the switching elements of Figure 24a.

35 In Figure 24c, instead of transistor pair T8/T9, a driver element (2402) is connected to the bus to improve signal quality.

09145139-082898

In Figure 24d, the external bus is configured as a unidirectional bus for better implementability in chips. Drivers 2403 are unidirectional, and there is no control signal S7.

Figure 25 proposes a possible synchronization circuit for Fig. 24. Registers 2401 for data storage are represented by 2501. Register 2502 is used for storing an RDY handshake signal, i.e., information that there is valid data on the bus or in 2501. If there is no valid data in 2501, output Q of 2502 is logical zero. If active data is received (RDY is active and logical 1), an enable signal (EN) is generated for registers 2501 and 2502 via AND-gate 2503, the data and RDY are stored with the rising edge of the clock pulse. The input of 2503 is inverting for output Q (of 2502)! If data is received again, the output Q (of 2502) is logical 1. The AND-gate delivers a logical 0 and the registers are not enabled via EN. If the data is forwarded over the bus, the activating signal of the output driver (OE) is used as Clear for 2502; Q of 2502 is logical 0 and new data can be stored with the next clock pulse. Lines Din and Dout are drawn as solid lines, since this is a bus system. 2501 is also drawn as a solid line, since the register corresponds to the width of the bus.

The registers can be configured as latches and coupled to the level of the clock signal (CLK) or the handshake (RDY). The circuit, however, becomes asynchronous, which results in considerable implementation problems and is associated with a non-negligible extra cost.

Figure 26 shows the test strategy to determine whether a connection can be set up, i.e., whether the network is free. The status information of switching elements 1, 2, 3, 4, 5, 6 which provides information on whether a

switching element is busy or free is located in a matrix 2603. The 90° switching elements 3, 4, 5, 6 form the corners, and the 180° switching elements 1 and 2 form the central elements and are doubled. In order to use a
5 switching element without collisions, the entire edge where the switching element is located must be free. For example, 1+2, 6+4, 3+5 can be used; however, 6+2, 6+1, 2+5, 2+4, 2+3, etc. cannot be used.

10 Thus, it must be ensured and tested that each edge is only occupied once. A register 2602, where the current configuration of the node is stored via input r, provides the corresponding data; so does routing table 2601, which forwards the data of the desired new bus to the matrix
15 via input t.

The test circuit is shown in Figure 26a. Each row (2605, 2606, 2607, 2608) is checked for the existence of a connection via an OR-gate (2609, 2610, 2611, 2612). If
20 there is a connected element in the row, the corresponding OR-gate delivers a logical 1. The result of the row is AND-gated with a connection to be newly set up in that row. If the row is occupied AND another connection is requested in the row, the respective AND-
25 gate delivers a logical 1. The outputs of all the AND-gates are OR-ed (2613). The result of the test thus delivers a logical 0 to 2604 if the requested connection is valid and a logical 1 if there is a collision.

30 The circuit of Figure 26 is only capable of processing one request per time unit. Time optimization is illustrated in Figure 27. Access requests from the buses to the circuit arrive via 2701. The routing table (2702) is made of a plurality of individual registers (2711)
35 instead of a common storage device. This allows the data of all the access requests to be read from the routing table via multiplexer 2703. The data of each access

request is forwarded to a matrix (2704), according to Figure 26, which receives the comparative data from the register with the current configuration of the node (2705). Circuit 2706 has an OR-gate, which determines the existence of a valid request to matrix 2704. The result of 2704 is AND-ed with the output of the OR-gate via an inverter. If there is an existing and valid access, a logical 1, otherwise a logical 0 is delivered as a result. Each matrix has a circuit 2706, whose results go to an arbiter 2707, which selects one of the valid accesses. Multiplexer 2708 is switched so that the data of the valid access goes to mixer 2709, which links the valid new access with the existing connection and forwards it to register 2705 for storage.

This circuit is capable of selecting one access out of four accesses. By changing the number of multiplexers (2703), matrices (2704), arbiter and multiplexer widths (2707, 2708), as well as the respective logics, one valid access out of any settable number of requests can be processed.

Often it is necessary to select more than one access out of a number of accesses. Lines 2801, 2805, 2802, and 2810, leading to an additional circuit (shown in Figure 28) and allowing the simultaneous selection of two accesses, serve this purpose. If 2810 is wired, line 2710 can be omitted. According to the principle presented below, any number of accesses can be simultaneously selected through cascading.

The information about which access has been selected as "valid" goes to decoder 2803 via 2801. The information is decoded so that only the access data of the access not selected is forwarded to the matrices via the three multiplexers 2804. By omitting the previously selected access, the number of matrices is reduced by one.

The decoder operates as shown in the following table:

70390
5

Decoder (2802)	MUX1	MUX2	MUX3
a	b	c	d
b	a	c	d
c	a	b	d
d	a	b	c

- 10 The table shows the "valid" bus selected via 2802 in the column "Decoder." Columns MUX1-MUX3 show which bus is selected by the respective multiplexer, depending on the value of 2802.
- 15 The matrices (2811), the logic (2806), and the arbiter (2807) operate as described in accordance with Figure 27. The data of the access selected by the arbiter is forwarded to mixer 2809 via multiplexer 2808. Mixer 2809, like mixer 2709, joins the data of the access selected by
- 20 the logic of Figure 28 to the output data of 2709, and forwards the generated access data to register 2705 via 2810. The input data of multiplexer 2808 must be picked up from the outputs of multiplexers 2804 due to their configuration.
- 25 The circuit according to Figure 28 can be cascaded deeper down by the principle described above with the number of matrices decreasing by one per cascade.
- 30 Figure 29 shows a register according to 2602 and 2705. The outputs of mixers 2709 and 2809 are sent as input data to the register via 2901. Each register bank 2902a/b manages one of the buses (B1, B2, ... Bm) of the node. The control of the node is stored in part a of a bank.
- 35 The timeout of the bus connection is defined in part b. Part b has a loadable counter, whose enable and reload

are settably selected by part a via multiplexer 2903.

T0400

Timeout Principle	Effect
BUS-ACK	The data transfers are counted. (Bus in use / packet size)
!BUS-ACK (inverted)	The clock pulses are counted with no data transfer. (Bus NOT is use / timeout)
en	Each pulse is counted / clock cycles
-	No timeout / permanent

10 The reload and enable signals of the counter are generated as follows:

T0401

Timeout Principle	reload (rld)	enable (en)
BUS-ACK	never	with data transfer
!BUS-ACK (inverted)	with data transfer	with no data transfer
en	never	always
-	never	never

20 The register needed for a reload of the counter, which contains the count set originally, is contained in 2902b. 2904 tests for count 0 to determine the timeout. 2904 is shown in the figure only for the sake of clarity; in the implementation the transmission signal (ripple carry) of the counter is used. The transmission clears the contents of 2902a, which then forwards the "bus free" status information and thus disconnects the bus. From the transmission, BUS-DISCONNECT is switched onto the bus as a signal and used for disconnecting the remaining bus segment. BUS-RECONF is sent with the data according to German Application No. DE 197 04 728.9 and, when it appears, also causes the bus to be disconnected. Both signals are sent to 2902 via OR-gate 2905 and cause the

register and the counter to be cleared. The timeout is de-activated by de-activating the enable signal in the previously mentioned tables, and the counter is loaded with a value greater than zero.

5

The data in the register are bus-oriented, not switching-element-oriented. This data goes via 2906 to mixers 2709 and 2809. Each control signal occurs m-fold (number of buses) and is denoted as $S_{i,m}$, where m is the bus and i is the number of the switching element. Prior to sending the data to a matrix according to Figure 26 or to a node according to Figures 23 and 24, these must be mapped so that there is only one sequence T_i . The mapping formula is $T_i = (S_{i,1} \cup S_{i,2} \cup S_{i,3} \dots \cup S_{i,m})$; in other words, all $S_{i,1}$ through $S_{i,m}$ are OR-ed. 2907 is responsible for this function and sends T to the matrices and switching elements via 2908.

10

15

20

25

30

35

Figure 30 shows an example of two bus connections. One node 3002 is assigned to configurable elements or groups of configurable elements (3001). Node 3003 transmits data to node 3004. The connection is set up statically. Node 3005 transfers the data dynamically to target node 3008. Segments 3006 and 3007 are busy, so that the direction of transmission changes. The X/Y spatial coordinates are provided in the nodes being traversed. The coordinates are kept unchanged, increased or decreased by one depending on the direction of travel. Using the numerical value of the coordinates, the direction of travel and the target can then be determined. The deviation in the direction of movement is calculated from the position (A, B, C, D) of the incoming bus at the node and the sign of the (X/Y)-movement. Compass points are used to identify the direction with y forming the north-south axis and x forming the east-west axis:

T0420

5

Direction of Travel	Movement in Compass Point Direction
$y = 0$	$x > 0 \Rightarrow E$ $x < 0 \Rightarrow W$
$y > 0$	N
$y < 0$	S
$x = 0$	$y > 0 \Rightarrow N$ $y < 0 \Rightarrow S$
$x > 0$	E
$x < 0$	W

09145139 002898

10

Using the direction of travel and the compass point direction of the incoming bus, it can be calculated which of the switching elements (1, 2, 3, 4, 5, 6) is being addressed. Both of the above calculations are known to those of ordinary skill in the related art, so that the required arithmetic unit (XY2ADR) can be configured using lookup tables. Thus the calculation is not described here in more detail, but reference is made to the above table.

15

The addressing in this example is relative.

20

Figure 31 shows the same example with absolute coordinates. Contrary to Figure 30, the coordinates in the nodes are not calculated, but compared with the node coordinates according to the higher-order coordinate system 3101. The connection is set up using comparisons greater than ($>$), less than ($<$), and equal to ($=$), for control. If both coordinates (X and Y) are equal to the node coordinates, the target has been reached; if one coordinate is equal to the node coordinate, the target axis of the coordinate has been reached.

25

30

The examples given in Figures 30 and 31 allow no deviation from the optimum direction. For example, if segment 3009 in Figure 30 were busy, the data could not

be further transferred. It is possible to allow a deviation from the predefined direction in the case of a busy segment, in which case the connection can be set up via 3010. however, the deviation allowance must be limited so as not to obtain unreasonable routing attempts. One reasonable limit for deviations from the predefined direction is ± 1 to ± 2 .

Figures 32a and 32b show the required periphery around a node 3201 to evaluate or modify the spatial coordinates.

Figure 32a modifies relative coordinates according to their direction of travel. Subtraction is performed in the positive direction (3203), and addition is performed in the negative direction (3202). The comparators (3204) check whether a coordinate has reached zero.

Figure 32b compares absolute coordinates with the node coordinates using comparator 3205.

In order to allow a deviation from the predefined direction, comparators 3205 and 3204 are expanded so that they check and forward the information of whether a coordinate is in the allowance range $(-deviation < coordinate < +deviation)$. Using this information, the arithmetic unit (XY2ADR) can modify the direction of travel within the limits of the allowable deviation in the event of a collision in the predefined direction and allow or prevent a deviation. This calculation is also trivial and may possibly be performed by expanding the lookup tables. In the following table, the maximum allowable deviation is given by A.

Direction of Travel	Movement in Compass Point Direction
$y-A = 0$	$x+A > 0 \Rightarrow E$ $x-A < 0 \Rightarrow W$

$y+A > 0$	N
$y-A < 0$	S
$x-A = 0$	$y+A > 0 \Rightarrow N$ $y-A < 0 \Rightarrow S$
$x+A > 0$	E
$x-A < 0$	W

The immediate surrounding of x and y is fuzzy, i.e., movements in opposite directions may be allowed, since $k-A < 0$ AND $k+A > 0$ may apply at the same time. This can be more narrowly defined if so desired by not allowing any direction of travel against the sign of k. If $k = 0$, all directions of travel are allowed.

Figure 33 shows the behavior of a segmented bus. The configuration of the drawing corresponds to the previous Figures showing a cell array and nodes. A transmitter node Sa sends data to receiver node Ea; another transmitter node Sb sends to Eb, and a last Sc sends to Ec, which is also receiver node Eb. Collisions occur on segments 3301 and 3302. To optimally utilize the bus (regardless of the fact that another path would be possible in principle), all the buses are set up to be quasi-permanent, with the exception of segments 3301 and 3302. These segments operate in the "registered" mode and arbitrate one of the buses in question according to the timeout set. The priority of the individual buses can be determined via the respective timeout. A more relevant bus receives higher "timeout rights," i.e., longer cycles, while a less relevant bus has only shorter cycles available.

The previous descriptions of the nodes in Figures 23-27 show, for the sake of clarity, only the interconnection in the directions of the four compass points. In fact, however, a configurable cell or a group of configurable

cells may also be connected to a node. The extensions needed therefore are shown in Figure 34. The symbols of the nodes correspond in Figure 34a to those in Figure 23, and those in Figure 34b to those in Figure 24. The connection of the configurable elements is denoted with Z. Matrix 2603 should be modified according to Figure 34c to become 3401. The interconnections are modified according to Figure 26.

10 The examples shown are illustrated as two-dimensional for the sake of clarity. In alternative embodiments, complex systems with any number of dimensions may be built in accordance with the present invention.

GLOSSARY OF TERMS

address lookup The address is not calculated, but generated by "lookup" in a memory.

5 **ALU:** Arithmetic and logic unit. Basic unit for processing data. The unit can perform arithmetic operations such as addition, subtraction, and also under some circumstances multiplication, division, series expansion, etc. The unit
10 may be designed as an integer unit or as a floating point unit. Likewise, the unit can perform logic operations such as AND, OR and comparisons.

setup phase: Cycle during which a bus segment is set up.

15 **arbiter:** Unit for distributing rights among signals.

bidirectional: Data transmission in both directions
(source/target1 <-> source/target2)

20 **broadcast:** Transmitting data of a PAE to multiple or all data receivers.

bus request: A request for a bus connection to be set up for data transfer (also called a connection request).

25 **bus segment:** Section of a bus system between two nodes.

bus states: The way a bus operates. Distinction is made between two main states:

30 **quasi-permanent:** The bus behaves like a continuous conductor. The bus can only be used by one data packet (until it is disconnected);

registered: A register to delay the data by one clock cycle is looped between each segment. Another data
35 packet can be arbitrated in each clock cycle (depending

on Timeout); Mixing the two states provides the status.

Segmented: Combines the advantages of the two states.

5

data receiver: The unit(s) that process(es) (further) the results of the PAE.

10

data transmitter: The unit(s) that make(s) data available as operands for the PAE.

data type: Type of data: characters, numerals, floating point numbers, signals (boolean), etc.

15

decimal decoder: Converts a binary signal into a decimal signal.

DFP: Data flow processor according to (unexamined) German Patent DE 44 16 881.

20

DISCONNECT: Signal generated by timeout counters/generators to disconnect a bus. It is sent to all the nodes of a bus.

25

DPGA: Dynamically configurable gate array. Known.

EALU: Expanded arithmetic and logic unit. ALUs expanded by special functions which are needed or are appropriate for operation of a data processing unit according to German Patent DE 44 16 881 A1. These are counters in particular.

30

35

elements: Collective term for all types of self-contained units that can be used as one piece in an electronic component. Elements thus include:

09145139-082890

- Configurable cells of all types
- Clusters
- RAM blocks
- Logic units
- 5 - Arithmetic and logic units
- Registers
- Multiplexers
- I/O pins of a chip

10 **enable:** Switching a register or a counter to an operational state.

FPGA: Field programmable gate array. Known.

15 **gate:** Group of transistors that carry out a basic logic function. Basic functions include, for example, NAND, NOR, transmission gates.

20 **transmission rate-optimized:** A bus system that is usually permanently set up has a high priority and is not affected by other accesses.

25 **H level:** Logical 1 level, depending on the technology used.

30 **Node:** Element that connects multiple bus segments together and actively controls the setup of the connection but is passive during the data transfer.

35 **configurable element:** A configurable element is an element of a logic unit that can be set for a specific function by a configuration string. Configurable elements are thus all types of RAM cells, multiplexers, arithmetic and logic units, registers and all types of internal and external interconnection descriptions, etc.

configurable cell: See logic cells.

configure: Setting the function and interconnection of a logic unit, a (FPGA) cell or a PAE (see reconfigure).

5

program loading unit (PLU): Unit for configuring and reconfiguring the PAE. Embodied by a microcontroller adapted specifically to this function.

10 **latch:** Storage element that normally relays a signal transparently during the H level and stores it during the L level. Latches are sometimes used in PAEs where the function of the levels is exactly reversed. An inverter is connected upstream from the clock pulse of a
15 conventional latch.

logic cells: Configurable cells used in DFPs, FPGAs and DPGAs, which perform simple logic or arithmetic tasks according to their configuration.

20

look-up table: Table that receives a value as an address and returns a result. For example, a number is given as an address and its sine is returned.

25

L level: Logical 0 level, depending on the technology used.

M-PLUREG: Register in which the interconnection of the PAE is set. The PLU writes into the register.

30

mask: A bit combination containing information concerning which signals of a source should be forwarded and which should be interrupted (masked out).

35

mixer: Unit that interconnects several signals according

to a certain mapping instruction.

- a) usually a logic operation (AND/OR) takes place, or
- b) the signals are combined to a bus, where
- c) several signal sources may be combined selectively via several multiplexers.

open-collector: Circuitry where the collector of a transistor is at a bus signal which is pulled to the H level by a pull-up. The emitter of the transistor is at ground. If the transistor switches, the bus signal is pulled to the L level. The advantage of the method is that a plurality of such transistors can control the bus without electrical collision. The signals are OR-linked, resulting in wired-OR.

PAE: Processing array element: EALU with O-REG, R-REG, R2O-MUX, F-PLUREG, M-PLUREG, BM UNIT, SM UNIT, sync UNIT, state-back UNIT and power UNIT.

partner node: Node with which a certain node has set up contact or wants to establish contact over a bus segment.

PLU: Unit for configuring and reconfiguring the PAE. Embodied by a microcontroller adapted specifically to this function.

priority type: The manner in which a priority identifier is analyzed.

priority decoder: The signal with the highest priority is relayed or enabled.

priority ID: Priority level information (high to low) of a bus connection.

priority logic: The signal with the highest priority is

relayed or enabled.

PullDown: Resistor that pulls a bus line to an L level.

5 **PullUp:** Resistor that pulls a bus line to an H level.

source-optimized: A bus system usually set up as registered and with low priorities so as to enable bus access by a number of data transmitters (sources).

10

spatial coordinates: Point information via a multidimensional coordinate system. **Absolute** or **relative** coordinates (relative distance from a reference point) can be used. Movement in the **positive direction** increases the numerical value of the coordinate; movement in the **negative direction** reduces the numerical value of a coordinate.

15

RECONFIG: Signal generated by configurable elements showing that the elements can be reconfigured and their activity has been completed. Used to disconnect all the buses involved and sent to all the nodes of a bus.

20

register bank: Combination of a number of registers of different sizes and functions into a group.

25

register bypass: Conductor to bypass a register and turn off the synchronization effect of the register.

30

RELOAD: Loading a counter with its initial value.

routing table: Table within a node containing information about connections to be set up.

35

round-robin arbiter: Arbiter that enables one signal

after the other in a sequence. The signal currently enabled receives the lowest priority and is then the last to be enabled again in the chain. The arbiter works in a circle.

5

busbar: Bus on which multiple bus segments are combined.

Schmitt trigger: Window comparator that assigns a signal one of two possible values thus improving signal quality.

10

switching table: A switching table is a ring memory which is addressed by a controller. The entries in a switching table can have any configuration strings. The controller can execute commands. The switching table responds to trigger signals and configures the configurable elements on the basis of an entry in a ring memory.

15

timeout generator: unit used to generate a timeout according to several criteria, such as

20

- clock cycles without successful connection setup;
- data packets transmitted;
- clock cycles;
- clock cycles with no transmission.

25

timeout: Something happens after a certain period of time (an operation is begun or interrupted).

timeout counter: See timeout generator.

30

gate: Switch that relays a signal or blocks it. Simple comparison: relay.

unidirectional: Data transmission in one direction (source -> target).

35

reconfigure: Reconfiguring any set of PAEs while any

remaining set of PAEs continue their own functions (see configure).

connection request: Request for a bus connection for data transfer (also called bus request).

cells: Synonymous with configurable elements.

target axis: the X/Y axis where X = 0 or X = axis, and Y = 0 or Y = axis is the target axis of X and Y, respectively.

state machine: Logic unit that can assume various states. The transitions between the states depend on various input parameters. These machines are used to control complex functions and are already known.

Name conventions

Module:UNIT

5 Mode of operation:MODE

Multiplexer:MUX

Negated signal:not

Register visible to PLU: PLUREG

Register, internal:REG

10 Shift register:sft

Function convention

NOT function !

15

I	Q
0	1
1	0

AND function &

20

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

25

OR function#, >

30

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

GATE function G

35

70550

5

EN	B	Q
0	0	-
0	1	-
1	0	0
1	1	1

09145139 " 082898